

REMARKS

This Amendment is in response to the Office Action dated January 29, 2003. Claims 1-14 are pending in the present application. Claims 15-55 have been withdrawn. Claims 1, 2, 8, 9 and 12 have been amended, and claims 56 and 57 are new. Accordingly, claims 1-14, 56 and 57 are pending in the present application.

Amended Claims

Applicants have amended independent claims 1 and 8 to clarify the present invention. In particular, claims 1 and 8 now recite “in response to a data manager call to locate a data identifier . . . *issuing a callback to the data manager*” and “continuing . . . an index-data fetch . . . if . . . the index manager receives a specific condition from the data manager *in response to the callback*.” Support for this amendment is found in the Specification at page 10, lines 2-14. No new matter has been presented.

Claims 2, 9 and 12 were amended to correct the numbering sequence and to provide correct antecedent basis. These amendments do not affect the scopes of claims 2, 9 and 12.

New Claims

Applicants have added claims 56 and 57 which depend on claims 1 and 8, respectively, and recite “specifying that a callback is required when the data identifier is located.” Support for this is found in the Specification at page 9, line 20 to page 10, line 2. No new matter has been presented.

35 U.S.C. § 102 Rejections

The Examiner rejected claims 1-14 under 35 U.S.C. §102(e) as being anticipated by

Mohan (U.S. Patent No. 6,009,425). In so doing, the Examiner stated:

With respect to claims 1 and 8, Mohan discloses in response to a data manager (abstract, col. 7, lines 34-38) call to locate a data identifier in an index (col. 8, lines 21-24), corresponding to a selected key value (col. 6, lines 32-34), performing the step of locating the data identifier in the index for the selected key value (); and continuing to carry out the index-data fetch for another data identifier (col. 8, lines 21-23), if there is another data identifier for the selected key value in the index (col. 6, lines 49-63, and the index manager receives a specific condition from the data manager (col. 8, lines 28-32).

Applicants respectfully traverse. The present invention is directed to a method for processing a database query. In accordance with the present invention, a data manager utilizes an index manager to locate a data identifier (data ID) in an index corresponding to a selected key value. Instead of returning the located data ID to the data manager, the index manager issues a callback and passes the located data ID to the data manager. The data manager then determines whether the data specified by the data ID will be returned to a runtime (e.g., if it satisfies a query predicate and/or is not consumed). If the specified data is *not* returned, the index manager receives such an indication from the data manager and continues the index-data fetch for the next qualifying data ID. (Specification, page 9, line 20 to page 10, line 13).

According to the preferred embodiment of the present invention, efficiencies are introduced to query processing where the queries result in repeated access to the index without data from data table being passed back to runtime. The index manager effectively treats the callback to data manager as the processing of an index predicate. The code path inherent in the data manager making repeated calls to index manager has been avoided. Furthermore, the data ID located initially by index manager need not be copied from the index, and the page containing the leaf node need not be released or unlatched, while the callback to data manager is occurring.

The present invention, as recited in claim 1, provides:

1. A method for processing a database query on a set of data in a database management system having a data manager and an index manager, the method comprising the steps of:

- a) in response to a data manager call to locate a data identifier in an index corresponding to a selected key value, performing the steps of:
 - i) locating the data identifier in the index for the selected key value; and
 - ii) issuing a callback to the data manager; and
- b) continuing to carry out the an index-data fetch for another data identifier if there is another data identifier for the selected key value in the index and the index manager receives a specific condition from the data manager in response to the callback.

Independent claim 8 is a computer product claim having similar scope to that of claim 1.

Mohan is directed to a system and method for performing record deletions in a RDBMS.

In Mohan, repeated traversals of an index tree by an index manager is avoided by storing and utilizing information in an Index Scan Cursor Control Block (ISCCB). (Abstract, col. 3, line 45 to col. 4, line 25). In addition, the information in the ISCCB is used to reduce the number of unnecessary lock calls. (Col. 11, lines 19-30).

Applicants respectfully submit that Mohan fails to teach or suggest “issuing a callback to the data manager” and “continuing to carry out an index-data fetch for another data identifier if . . . the index manager receives a specific condition from the data manager in response to the callback,” as recited in claims 1 and 8. According to the present invention, the index manager issues *a callback* to the data manager instead of *returning* the data ID to the data manager. The index manager then waits to receive a specific condition from the data manager before it continues the index-data fetch.

In contrast to the present invention, Mohan discloses that the data manager invokes the index manager to locate a record identifier that satisfies a query. Once located, the index manager *returns* the record identifier to the query processor so that the corresponding record can be modified (e.g., deleted). (FIG. 5A, step 104). At no point does Mohan teach or suggest

“issuing a callback to the data manager” in response to a call from the data manager to locate a data identifier in the index, as recited in claims 1 and 8.

Moreover, Mohan fails to teach or suggest “continuing to carry out an index-data fetch for another data identifier if . . . the index manager receives a specific condition from the data manager in response to the callback,” as recited in claims 1 and 8. In Mohan, once the record identifier is returned to the query processor, the query processor directs the data manager to delete the record. Thereafter, the data manager requests the index manager to delete the record identifier (key), which it does, then the index manager updates the ISCCB. (FIG. 5A and 5B, steps 104-120). Another record identifier is not sought until after the data manager returns to the query processor and the query processor issues *another* call (FIG. 5C, step 124). Only then does the index manager obtain the ISCCB to locate the next record identifier. In contrast to the present invention, the index manager *does not* continue “to carry out an index-data fetch for another data identifier if . . . the index manager receives a specific condition from the data manager in response to the callback,” as recited in claims 1 and 8.

Applicants respectfully submit that Mohan fails to teach or suggest the combination of elements recited in claims 1 and 8. Accordingly, claims 1 and 8 are allowable over the cited references. Claims 2-7, 9-14, 56 and 57 depend on claims 1 and 8, respectively, and the above arguments apply with full force. Therefore, claims 2-7, 9-14, 56 and 57 are also allowable over the cited reference.

Conclusion

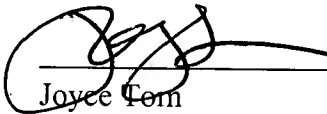
In view of the foregoing, it is submitted that the claims 1-14, 56, and 57, as now presented, are allowable over the cited reference and are in condition for allowance. Applicants respectfully request reconsideration of the rejections and objections to the claims, as now

presented.

Applicants believe that this application is in condition for allowance. Should any unresolved issues remain, Examiner is invited to call Applicants' attorney at the telephone number indicated below.

Respectfully submitted,

April 28, 2003
Date


Joyce Tom
Sawyer Law Group LLP
Attorneys for Applicant(s)
Reg. No. 48,681
(650) 493-4540